Artificer Pro

Documentation

[incomplete]

Getting Started

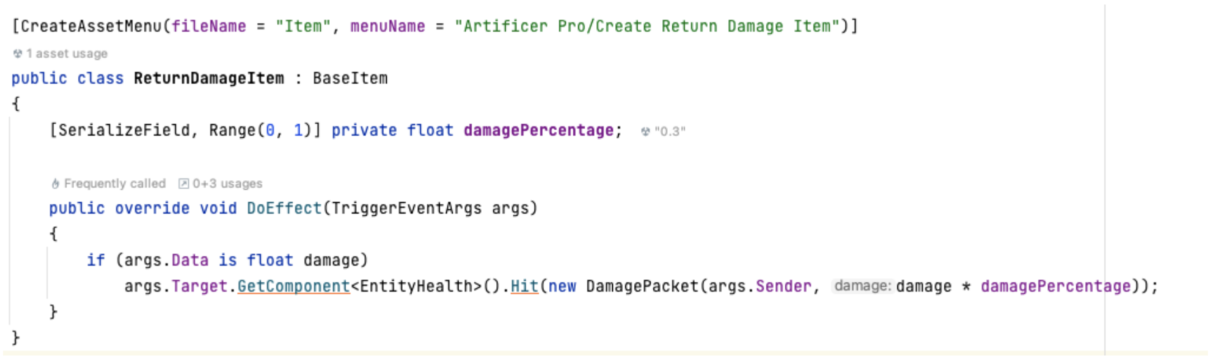
Demo

There is a demo included in Assets/Wayfarer Games/Artificer Pro/Demo. I strongly recommend you poke around that folder before doing anything else - it has a simple top down shooter project (using the old input system for input).

Items

Each item is a scriptable object. Extend the BaseItem class and override the DoEffect function to trigger effects! **Use [CreateAssetMenu] to add it to the right click -> create menu in Unity**.

Here is a simple "return damage" item type, which would be useful for an effect similar to Minecraft's "Thorns" enchantment.



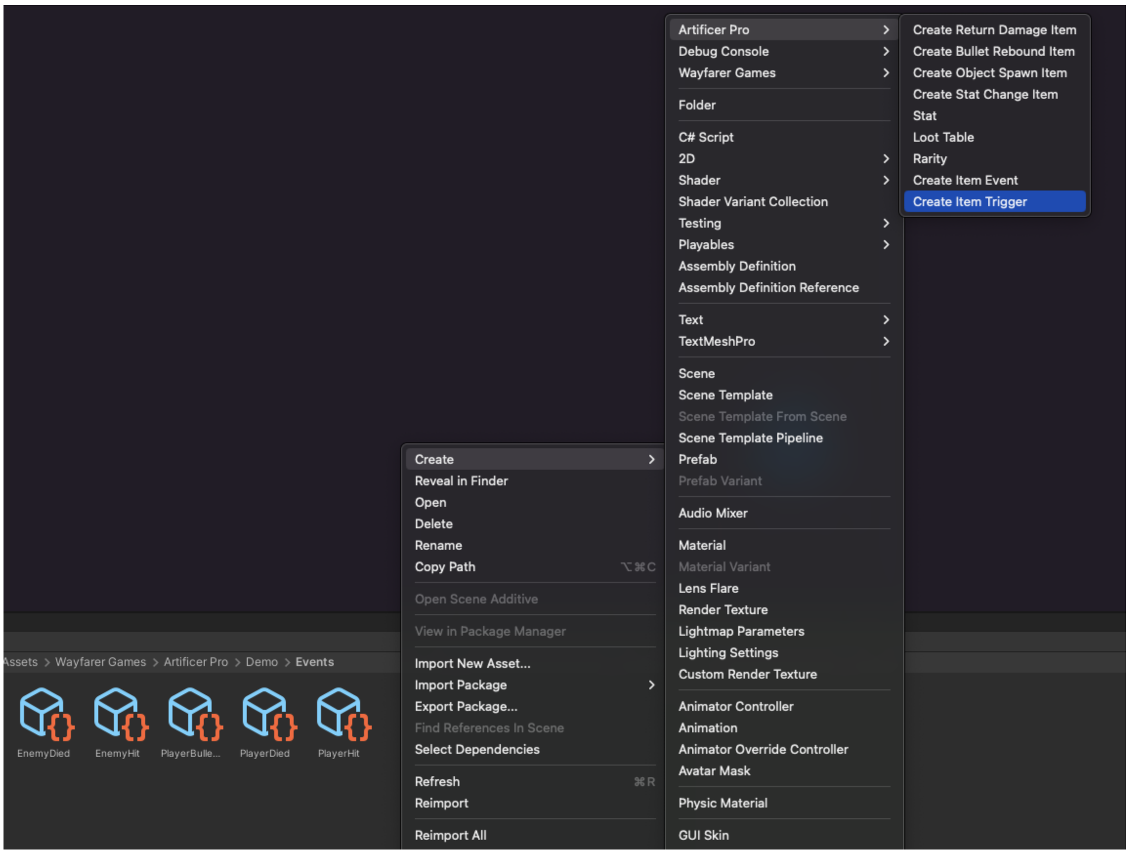
```
[CreateAssetMenu(fileName = "Item", menuName = "Artificer Pro/Create Return Damage Item")]
public class ReturnDamageItem : BaseItem
{
    [SerializeField, Range(0, 1)] private float damagePercentage;

    public override void DoEffect(TriggerEventArgs args)
    {
        if (args.Data is float damage)
            args.Target.GetComponent<EntityHealth>().Hit(new DamagePacket(args.Sender, damage * damagePercentage));
    }
}
```

In the DoEffect function, you have access to the "TriggerEventArgs" object. This will give you a Target (in this case, the enemy that dealt the damage), a Sender (in this case, the player), and an object Data value (in this case, a float for the amount of damage dealt). This is passed through from any Triggers that you create.

Triggers

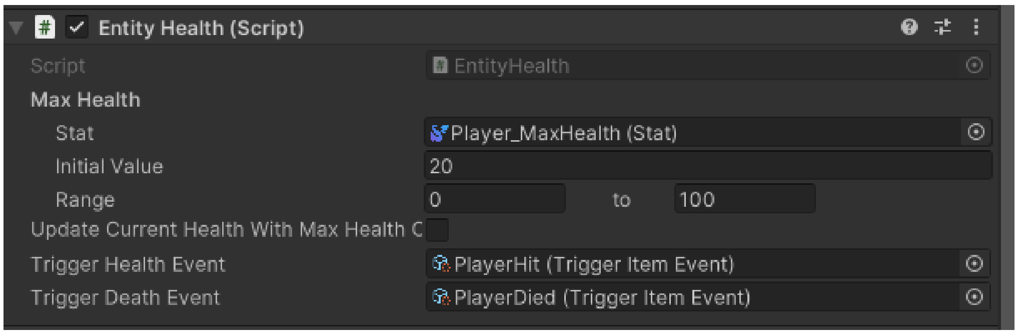
The backbone of the powerup system is the Trigger system. Create new item triggers in the project like so:



It's **up to you** to call these triggers in your code! In the demo, there's an example of this in the EntityHealth class:

```
[SerializeField] private TriggerItemEvent triggerHealthEvent;
[SerializeField] private TriggerItemEvent triggerDeathEvent;
```

Assign the Trigger scriptable object in the inspector, like so:



Entity Health (Script)

Script: EntityHealth

Max Health
- Stat: Player_MaxHealth (Stat)
- Initial Value: 20
- Range: 0 to 100

Update Current Health With Max Health C

Trigger Health Event: PlayerHit (Trigger Item Event)

Trigger Death Event: PlayerDied (Trigger Item Event)

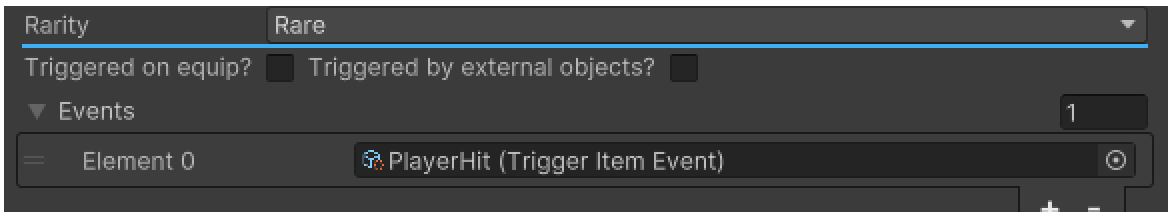
Then, when you want the Trigger to happen, call trigger.Invoke, passing through a new TriggerEventArgs, which contains the Sender, the Target, and any extra data you want to send:

```
public void Hit(DamagePacket damage)
{
    _currentHealth -= damage.Damage;
    triggerHealthEvent.Invoke(new TriggerEventArgs (damage.Sender,  target: gameObject, damage.Damage));

    if (_currentHealth <= 0)
        triggerDeathEvent.Invoke(new TriggerEventArgs (damage.Sender,  target: gameObject, damage.Damage));

    if (_currentHealth >= MaxHealth)
        _currentHealth = MaxHealth;
}
```

Your Item code will be listening out for these - make sure you assign the correct Triggers on the Item object, too.



Rarity: Rare

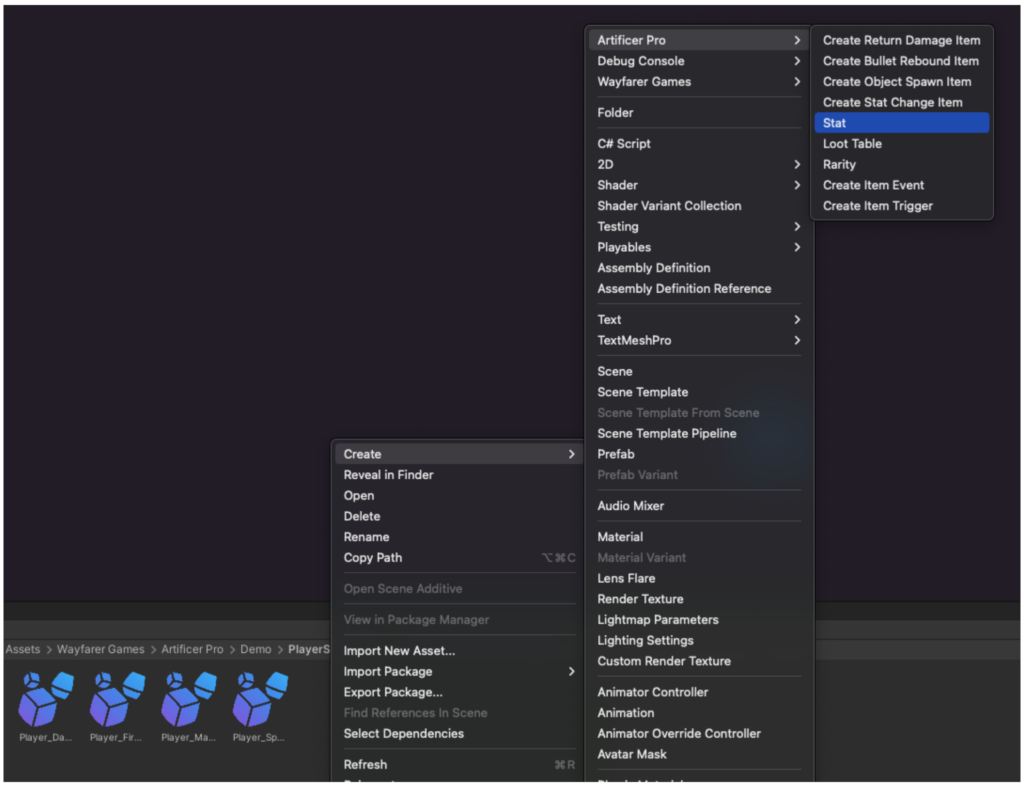
Triggered on equip? Triggered by external objects?

Events: 1
- Element 0: PlayerHit (Trigger Item Event)

Stats

Included in Artificer Pro is a robust stats system, which can be used to create a wide variety of effects. You'd use this for things like speed, max health, resistance, fire rate, damage multipliers, etc. Again, the Entity Health class in the demo shows exactly how this works, so it's worth having a poke around there.

You can create new stats like so:



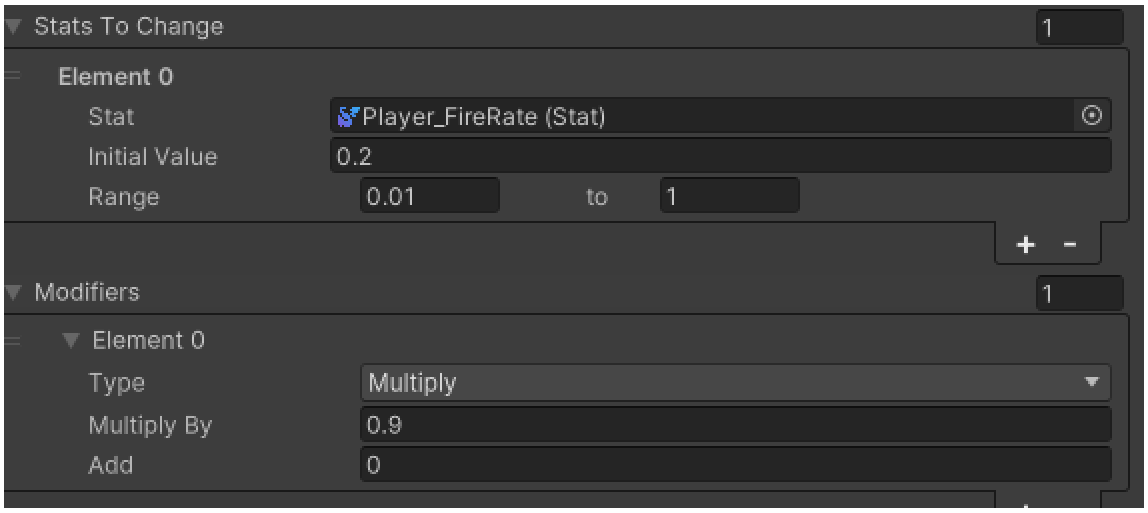
To use a stat in your code, reference it in your code like so:

```
[SerializeField] private Stat maxHealth;
```

You get the value of the stat with stat.GetValue(gameObject) - passing through the current game object to allow for other objects to use the stat. I usually make an accessor function like this, so I can use "MaxHealth" in the code to get the final value.

```
3 usages
public float MaxHealth => maxHealth.GetValue(gameObject);
```

Stats can be changed during the game using StatModifiers:

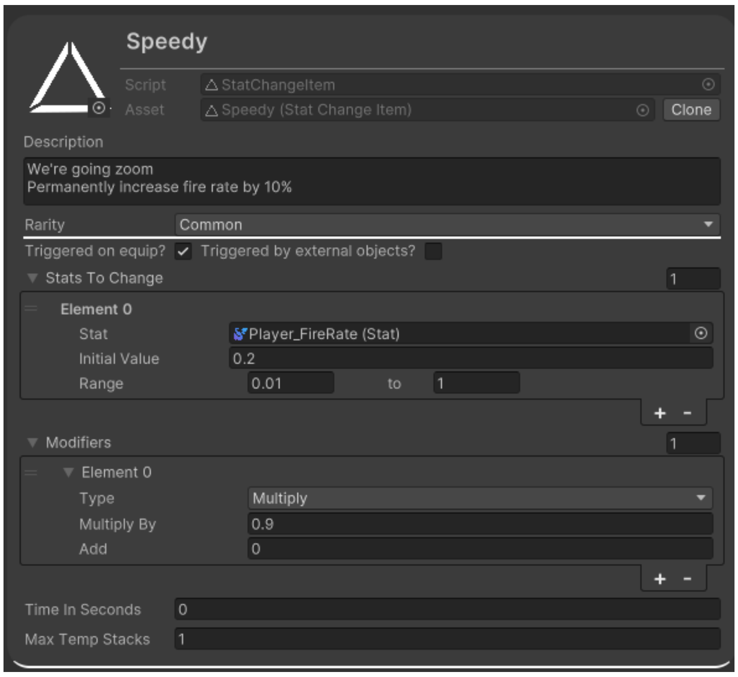


```
Stats To Change                                   1
  Element 0
    Stat            Player_FireRate (Stat)
    Initial Value   0.2
    Range           0.01        to      1
                                                 + -
Modifiers                                         1
  Element 0
    Type            Multiply
    Multiply By     0.9
    Add             0
```

```
statsToChange[i].AddModifier(modifiers[i], target);


statsToChange[i].RemoveModifier(modifiers[i], target);
```

In the provided demo, there is an Item type for changing stats based on events - you'll be able to do things like "increase fire rate for 10 seconds when an enemy dies" with zero code using this. It has options for permanent increases, temporary stacks that decay after a certain amount of time has passed.



```
Speedy
Script    StatChangeItem
Asset     Speedy (Stat Change Item)                 Clone
Description
We're going zoom
Permanently increase fire rate by 10%
Rarity                Common
Triggered on equip? ✓  Triggered by external objects?
Stats To Change                                   1
  Element 0
    Stat            Player_FireRate (Stat)
    Initial Value   0.2
    Range           0.01        to      1
                                                 + -
Modifiers                                         1
  Element 0
    Type            Multiply
    Multiply By     0.9
    Add             0
                                                 + -
Time In Seconds     0
Max Temp Stacks     1
```